“Introducción a la Administración de Proyectos”:

Objetivo del curso

- Adquirir **técnicas y conocimientos** para administrar proyectos eficientemente.
- Optimizar recursos aplicando las **mejores prácticas del PMI** (Project Management Institute).

Temario principal

- Guía del PMBOK (Project Management Body of Knowledge)
- Prácticas con MS Project
- Casos de estudio
- Presentaciones y reflexiones
- Examen y mediciones

Conceptos clave

¿Qué es un proyecto?

- **Esfuerzo temporal** para crear un **producto, servicio o resultado único.**
- Implica **recursos organizados**, es de **duración limitada**, y produce un **resultado con valor**.

¿Qué es administración de proyectos?

- Aplicación de **conocimientos, habilidades, herramientas y técnicas** para cumplir con los requerimientos de un proyecto.

¿Qué es el PMI?

- Asociación sin fines de lucro (desde 1969).
- Profesionaliza la administración de proyectos.

- Dueño del **PMBOK**, certifica profesionales y estandariza prácticas.

¿Qué es el PMBOK?

- Cuerpo de conocimiento aceptado globalmente.
- Establece:
  - **Entorno del proyecto** (estructura, organización).
  - **Ciclo de vida** y **procesos** del proyecto.
  - **9 áreas de conocimiento.**
  - **Lenguaje común** para proyectos.

Las 9 áreas del conocimiento (PMBOK)

1. **Integración:** Coordinar todos los elementos del proyecto.
2. **Comunicación:** Gestión de información durante todo el proyecto.
3. **Alcance:** Definir y controlar qué incluye y qué no incluye el proyecto.
4. **Calendario (Tiempo):** Orden y secuencia de actividades.
5. **Costo:** Control y asignación del presupuesto.
6. **Calidad:** Cumplir con los requisitos del proyecto y mejora continua.
7. **Riesgos:** Identificación y gestión de eventos externos que puedan afectar.
8. **Recursos humanos:** Organización y gestión del equipo de trabajo.
9. **Aprovisionamiento (Recursos materiales):** Compra y administración de bienes/servicios necesarios.

Problemas comunes en proyectos

- Cambios en requerimientos
- Poco involucramiento de usuarios
- Débil conocimiento técnico del equipo
- Uso inadecuado de métodos y herramientas
- Expectativas poco realistas
- Falta de apoyo gerencial
- Mala comunicación y planificación

Estadísticas sobre fracasos de proyectos

- **Chaos Report (1994):** solo 16.2% de los proyectos fueron exitosos.
- **OASIG (1995):** tasa de éxito entre 20%-30%.
- **KPMG Canadá (1997):** 61% considerados fracasados.
- **Conference Board (2001):** 40% no cumplen expectativas.
- **Robbins-Gioia (2001):** 51% de implementaciones ERP fallidas.

Introducción a los Sistemas de Información

- La **revolución tecnológica** ha transformado hogares, oficinas y empresas.
- Se integran computadoras y sistemas de información a la **estrategia empresarial**.
- El director de informática ahora tiene un rol más de **gestor estratégico** que técnico.

Importancia del uso de tecnologías de información

- Brindan **ventajas competitivas** y **estratégicas**.
- El directivo debe tener una **visión global** del negocio y los sistemas.

¿Qué es un Sistema de Información (SI)?

> "Conjunto de componentes interrelacionados que recopilan, procesan, almacenan y distribuyen información para soportar la toma de decisiones y el control." – *Laudon & Laudon (2004)*

Esquema funcional:

- **Entrada → Procesamiento → Salida → Retroalimentación**
- Influido por el ambiente (clientes, proveedores, competidores, etc.)

Conceptos clave

Dato:

- Representación simbólica sin valor por sí sola.

Información:

- Datos organizados con sentido, útiles para decisiones.

Usuario:

- Interactúa con el sistema, genera entradas y recibe salidas.

Componentes funcionales del SI

- **Entrada:** datos ingresados (por humanos o dispositivos).
- **Procesamiento:** clasificación, cálculo, validación.
- **Salida:** resultados, gráficos, reportes.
- **Retroalimentación:** valor añadido para mejorar decisiones o el sistema.

Perspectiva de negocio (Andreu y Ricart, 1996)

- El SI recopila, elabora y distribuye información relevante para:
  - Operación
  - Dirección y control
  - Toma de decisiones alineadas con la estrategia

Tipos de Sistemas de Información

Sigla	Sistema	Función principal
TPS	Transaction Processing System	Apoyo en actividades diarias; registra transacciones
OAS	Office Automation System	Mejora productividad de tareas administrativas
KWS	Knowledge Work System	Apoyo a trabajadores especializados (como arquitectos)

Sigla	Sistema	Función principal
MIS	Management Information System	Reportes para gerentes sobre operaciones regulares
DSS	Decision Support System	Apoyo en decisiones semi/no estructuradas
GDSS	Group DSS	Decisiones grupales colaborativas, entradas anónimas
ES	Expert System	Simula decisiones de expertos humanos
EIS	Executive Information System	Información crítica para ejecutivos con gráficos y seguridad

---

Características destacadas

- **TPS:** volumen alto, repetitivo, necesita precisión.
- **MIS:** reportes programados, por demanda o por excepción.
- **DSS:** análisis “¿Qué pasa si…?”, comparación y gráficos.
- **GDSS:** fomenta participación sin sesgos, soporte flexible.
- **EIS:** visual, rápido, seguro, remoto, personalizado.

Gestión del alcance

Implica todos los procesos necesarios para garantizar que el proyecto incluya todo el trabajo necesario (y solo ese trabajo) para completarse con éxito. Incluye: iniciación, planificación del alcance, definición, verificación y control de cambios.

¿Qué es el alcance del proyecto?

Es la suma de todos los productos y servicios que se deben entregar en un proyecto. Define qué está incluido y qué no. Todos los involucrados deben compartir una visión común de los entregables y las tareas necesarias para lograrlos. Los objetivos deben cumplir con el criterio SMART (específicos, medibles, alcanzables, realistas, con tiempo definido).

Alcance de producto vs alcance de proyecto

- **Alcance de producto:** se refiere a las características y funciones requeridas del producto o servicio.
- **Alcance de proyecto:** se refiere al trabajo necesario para entregar dicho producto con las características solicitadas.

Procesos de gestión del alcance

Incluyen: iniciación, planificación, definición, verificación y control de cambios del alcance.

Iniciación

Proceso que autoriza formalmente un nuevo proyecto o fase. Garantiza soporte organizacional, asigna un jefe de proyecto, alinea el proyecto con los objetivos estratégicos y asegura recursos.

Métodos para seleccionar un proyecto

Se seleccionan los mejores proyectos con base en:

- Necesidades generales de la organización
- Categorización
- Análisis financiero
- Métodos de puntuación

Acta del proyecto (Project Charter)

Documento que formaliza un proyecto y establece objetivos, criterios de éxito, restricciones y roles. Describe el producto, necesidades de negocio y otorga autoridad al jefe de proyecto. Debe ser emitido por un directivo con autoridad suficiente.

Planificar el alcance

Consiste en desarrollar un enunciado escrito que servirá como base para futuras decisiones y acuerdos con el cliente.

Enunciado del alcance

Documento que genera un entendimiento común del alcance. Debe incluir:

- Justificación del proyecto
- Producto y entregables
- Objetivos (SMART)

Plan de gestión del alcance

Define cómo se gestionará el alcance y sus cambios. Es parte del plan global del proyecto. Establece cómo se identificarán, clasificarán e integrarán los cambios en el proyecto.

Definir el alcance

Consiste en dividir el trabajo en partes manejables para:

- Mejorar la precisión de estimaciones
- Medir desempeño
- Definir responsabilidades

Entregables

Todo producto, resultado o elemento medible, tangible y verificable que se debe entregar. Un **work package** es el entregable más pequeño dentro de la estructura del proyecto.

Work Breakdown Structure (WBS / EDT)

Herramienta clave para la planificación. Descompone jerárquicamente los entregables del proyecto en componentes más pequeños. Representa la totalidad del trabajo y permite controlar y calendarizar cada paquete de trabajo.

Reglas para crear una WBS

- Cada elemento representa un entregable medible
- Cada nivel debe sumar los elementos inferiores
- Un elemento solo puede tener un padre
- Los entregables deben dividirse de forma lógica
- Deben ser únicos y descomponerse hasta el nivel necesario para planificar y controlar

---

Nivel de detalle de la WBS

Debe ser suficiente para un control efectivo. No debe ser tan pequeño que haga costoso el control ni tan grande que se pierda precisión. Todos los entregables, incluso los de gestión y comunicación, deben estar incluidos.

Métodos de creación de la WBS

- Por fases del ciclo de vida
- Por entregables principales
- Por subproyectos (por ejemplo, contratistas)

Productos adicionales de la WBS

- Diccionario del EDT: contiene ID, descripciones, calendario, recursos, costos estimados, calidad, criterios de aceptación, referencias técnicas.

OBS y RAM

- **OBS (Organizational Breakdown Structure):** descomposición jerárquica de organizaciones responsables del proyecto.
- **RAM (Responsibility Assignment Matrix):** matriz de asignación de responsabilidades.

Verificar el alcance

Proceso para obtener la aceptación formal de los entregables. Requiere revisar los entregables y sus resultados, junto con el control de calidad. Las inspecciones se usan para verificar cumplimiento (revisiones, auditorías, ensayos).

Resultados de la verificación del alcance

- Aceptación formal de entregables (documentada)
- Solicitudes de cambio en caso de no aceptación
- Actualización de la documentación del proyecto

Controlar el alcance

Monitorear el estado del proyecto y administrar cambios a la línea base del alcance. Asegura que los cambios pasen por un proceso formal. Evita el “scope creep” (aumento no controlado del alcance), considerando que el cambio es inevitable y debe gestionarse.

Diferencias entre usar y no usar bases de datos

Antes de las bases de datos, los programas estaban ligados directamente a sus propios archivos de datos. Esto causaba redundancia, inconsistencia de la información y pérdida de confianza en los sistemas. Aún hoy, muchas empresas enfrentan problemas similares por depender de documentos individuales y hojas de cálculo sin un sistema unificado.

Origen del enfoque de bases de datos

El enfoque surge en los años 60 como una forma de describir y organizar los datos de una empresa de forma integral. En vez de que cada usuario cree su propia versión, se busca un diseño único, con mínima redundancia y máxima cobertura de necesidades.

¿Qué es una base de datos?

Es una colección de archivos **interrelacionados**, diseñada cuidadosamente para evitar la duplicidad y cubrir las necesidades de la empresa. Los archivos contienen múltiples registros de un mismo tipo (ej. alumnos, cursos, inscripciones), pero están estructurados y controlados bajo un sistema común.

¿En qué casos conviene usar bases de datos?

Cuando hay:

- Información distribuida en varios archivos interrelacionados
- Necesidad de acceso por múltiples usuarios o aplicaciones
- Grandes volúmenes de datos (como padrones electorales o directorios telefónicos)

Diseñar una base de datos con un DBMS se vuelve la opción más conveniente.

¿Qué es un sistema de gestión de base de datos (DBMS)?

Es un conjunto de programas y estructuras que permiten almacenar, consultar, actualizar y proteger datos. Su propósito es ofrecer un entorno **eficiente y accesible** para manipular información relevante de una organización.

Características y funciones de un DBMS

- Manejo de grandes volúmenes de datos
- Definición de estructuras de almacenamiento
- Mecanismos de recuperación y seguridad ante fallos
- Soporte para múltiples usuarios sin conflictos
- Acceso eficiente, sin necesidad de saber detalles técnicos

Objetivos de los sistemas de base de datos

Resolver las limitaciones de los sistemas tradicionales de archivos:

- **Redundancia e inconsistencia:** múltiples copias de datos con posibles contradicciones
- **Dificultad de acceso:** nuevos requerimientos implican escribir nuevos programas
- **Aislamiento de datos:** estructuras diferentes hacen complejo desarrollar aplicaciones
- **Acceso concurrente inseguro:** múltiples usuarios pueden provocar inconsistencias si no se controla
- **Falta de seguridad:** no todos los usuarios deben ver toda la información
- **Falta de integridad:** no se aplican automáticamente restricciones (por ejemplo, saldos mínimos)

Problemas del procesamiento de archivos tradicionales

A medida que se agregan programas y archivos sin planificación central, se acumulan problemas como:

- Repetición innecesaria de datos
- Resultados inconsistentes
- Dificultad para realizar consultas nuevas
- Alto costo de mantenimiento

Rol del gestor de base de datos (DBMS)

Es el módulo que se encarga de la interfaz entre los datos almacenados y los programas o consultas. Es responsable de:

- **Interacción con el sistema de archivos:** traducir comandos de alto nivel en operaciones físicas
- **Implantación de integridad:** verificar que los datos cumplan reglas y restricciones
- **Implantación de seguridad:** controlar el acceso según roles de los usuarios
- **Copia de seguridad y recuperación:** restaurar el estado previo ante fallos del sistema
- **Control de concurrencia:** coordinar accesos simultáneos sin generar errores o duplicidad

---

Consideraciones técnicas

Dado que las bases de datos pueden ocupar gigabytes o terabytes, es esencial que el sistema minimice los movimientos entre disco y memoria para mantener un buen desempeño. Además, el sistema debe equilibrar eficiencia, almacenamiento y tiempos de respuesta, ofreciendo vistas simples para los usuarios sin exponer la complejidad interna.

Introducción al diseño de bases de datos relacionales

El objetivo principal del diseño de una base de datos relacional es crear esquemas de relaciones (tablas) que minimicen la redundancia y faciliten la recuperación eficiente de la información. Para lograr esto se utilizan formas normales y el análisis de dependencias de los datos, las cuales reflejan reglas y limitantes propias del modelo del mundo real que se quiere representar.

Problemas comunes en bases de datos mal diseñadas

- Redundancia de datos: una persona aparecería varias veces si tiene más de un coche.
- Inconsistencia: modificar un dato (como un nombre) implica revisar múltiples registros.
- Presencia de valores nulos innecesarios: genera desperdicio de espacio.
- Dificultad en operaciones básicas: inserción, eliminación y modificación se vuelven complejas.
- Mayor consumo de almacenamiento: por mala agrupación de atributos en esquemas no optimizados.

Fases del diseño de bases de datos

Recolección y análisis de requerimientos: entrevistas y documentación de necesidades de los usuarios.

Diseño conceptual: creación de un modelo abstracto (Modelo Entidad-Relación) que describe entidades, relaciones y atributos.

Diseño lógico: transformación del modelo conceptual al modelo de datos del SGBD (normalmente modelo relacional).

Diseño físico: especificación de estructuras de almacenamiento y organización interna de archivos.

Modelo Entidad-Relación (ER)

Propuesto por Peter Chen en 1976, permite representar de forma gráfica los objetos del mundo real que queremos modelar, sus relaciones y atributos. Es ampliamente aceptado y utilizado en herramientas CASE.

Entidades

Son objetos del mundo real con existencia propia, sobre los cuales queremos guardar información. Se representan con rectángulos y su nombre debe ser un sustantivo plural.

Asociaciones

Representan acciones o relaciones entre dos o más entidades. Se dibujan como rombos y generalmente se nombran con verbos. Pueden incluir atributos propios, como por ejemplo, la fecha de una venta.

Grado y cardinalidad de una asociación

- **Grado:** número de entidades involucradas (normalmente 2, relaciones binarias).
- **Cardinalidad:** cantidad de elementos relacionados entre entidades. Tipos más comunes: 1:1, 1:N, N:N. Se representan con notaciones sobre las líneas del diagrama.

Tipos de participación

- **Opcional (parcial):** una entidad puede no participar en una relación.

- **Obligatoria (total):** todas las instancias de una entidad deben estar relacionadas.

Atributos

Características o propiedades de entidades o asociaciones. Se representan con elipses. Se clasifican en:

- **Simples o compuestos:** indivisibles o formados por otros atributos.
- **Mono valuados o multivaluados:** uno o varios valores por entidad.
- **Almacenados o derivados:** almacenados directamente o calculados a partir de otros.

Identificador de identidad

Es uno o varios atributos que identifican de forma única a cada instancia de una entidad (como matrícula o número de cliente). Se subraya en los diagramas para distinguirlo.

Metodología para construir un modelo ER básico

1. Identificar entidades (normalmente sustantivos).
2. Agregar atributos a entidades.
3. Determinar identificadores únicos.
4. Identificar asociaciones (normalmente verbos).
5. Determinar la cardinalidad.
6. Incluir atributos en asociaciones si es necesario.
7. Verificar con requerimientos del sistema.
8. Refinar el modelo.

Modelo ER extendido

Incluye elementos adicionales para modelar situaciones más complejas:

Roles: se usan cuando una entidad participa en más de una relación o se relaciona consigo misma.

Entidades generalizadoras y especializadoras (relaciones ISA): permiten representar jerarquías. Una superclase agrupa atributos comunes y las subclases heredan o añaden atributos específicos. Se representan con triángulos. Puede ser:

- **Disjunta:** una entidad pertenece solo a una subclase.
- **Solapada:** una entidad puede pertenecer a más de una subclase.

Entidades fuertes y débiles

- **Fuertes:** tienen existencia propia y una clave primaria.
- **Débiles:** dependen de otra entidad para su identificación y existencia. Se representan con doble trazo.

Restricciones de integridad

Existen dos tipos:

Implícitas: derivadas naturalmente del modelo. Ejemplo: una relación solo es válida si existe una pareja proveedor-material en la realidad.

Adicionales: no se muestran en el diagrama, pero son esenciales para el funcionamiento. Se definen mediante reglas lógicas o condiciones, como:

- Un curso no debe iniciar después de su fecha final.
- El sueldo de un empleado debe ser menor al de su jefe.
- El valor de una medición debe estar dentro de límites definidos.

Restricciones basadas en recursos

Son guías útiles para definir integridad adicional en los modelos:

- Recursos humanos: capacidad, legalidad, distribución de salarios.
- Recursos ecológicos: emisión de desechos, normas ambientales.
- Tiempo: ventanas válidas para acciones o uso de recursos.
- Instalaciones: capacidad, disponibilidad, logística.
- Transporte: unidades disponibles, combinaciones de carga.

- Recursos financieros y materiales: presupuestos, tiempos de surtido, límites operativos.

Introducción

Para implementar un Modelo Entidad-Relación (MER) en una base de datos relacional, se traduce el modelo a un conjunto de tablas (Modelo Relacional - MR). Las tablas se definen con una notación en la que las columnas subrayadas indican la llave primaria, que garantiza la unicidad de cada fila.

Procedimiento de transferencia

Por cada **entidad**, se define una tabla con columnas iguales a sus atributos. El identificador de la entidad se convierte en la llave primaria de la tabla. Si no existe un identificador natural, se debe crear uno artificial.

Por cada **asociación N:N**, se crea una tabla con las llaves primarias de las entidades participantes y los atributos propios de la asociación. La llave primaria es la combinación de las llaves de ambas entidades, aunque puede añadirse una clave artificial por eficiencia.

Por cada **asociación 1:N**, se agrega la llave primaria de la entidad del lado 1 como columna en la tabla del lado N. No se crea una nueva tabla.

Por cada **asociación 1:1**, basta con añadir la llave primaria de una entidad en la tabla de la otra.

Ejemplo de aplicación del procedimiento

Dado un MER con entidades A, B y C:

- A(a1, a2, a3)
- B(b1, b2)
- C(c1, c2, c3, c4)

Si existe una asociación X de tipo N:N entre A y C:

- X(a1, c1, x1, x2)

Si existe una asociación Y de tipo 1:N entre A y B, se modifica B:

- B(b1, b2, a1)

Resultado final:

- A(a1, a2, a3)
- B(b1, b2, a1)
- C(c1, c2, c3, c4)
- X(a1, c1, x1, x2)

Reglas para elementos adicionales del MER

Relaciones ISA:
Las relaciones ISA son 1:1. Las subclases heredan la llave primaria de la superclase. Ambas comparten la misma clave.

Ejemplo:

- G(g1, g2, g3)
- Ea(g1, a1, a2)
- Eb(g1, b1)

Entidades fuertes y débiles:
Las entidades fuertes heredan su clave a las débiles. La tabla de la entidad débil incluye la clave de la fuerte más un atributo adicional para distinguir entre múltiples relaciones.

Ejemplo:

- F(f1, f2)
- D(f1, d1, d2)

Roles:
Cuando se usan roles por relaciones reflexivas o múltiples relaciones entre entidades, se sigue el algoritmo general, pero se renombran las columnas para evitar ambigüedad.

Ejemplo:

- E(e1, e2)
- R(e1, RolDeEe1, r1, r2)
  (R indica que la relación es entre dos elementos de E, distinguidos por el rol)

Comunicación

Es el conjunto de procesos necesarios para recopilar, distribuir, almacenar, recuperar y eliminar información del proyecto. Una comunicación eficaz conecta a los interesados superando diferencias culturales, organizacionales y de nivel técnico, asegurando la alineación durante toda la ejecución del proyecto.

Dimensiones de la comunicación

Puede clasificarse como:

- Interna y externa
- Formal e informal
- Vertical y horizontal
- Oficial y no oficial
- Escrita, oral, verbal y no verbal

Ejemplos de habilidades de comunicación

- Escucha activa
- Preguntar y explorar ideas
- Educación e investigación de datos
- Administración de expectativas
- Persuasión, negociación, resolución de conflictos
- Parafrasear, resumir, proponer acciones

Procesos de comunicación según el PMBOK

10.1 Identificar a los interesados
10.2 Planear la comunicación
10.3 Distribuir información

10.4 Administrar las expectativas
10.5 Reportar el desempeño

Identificar a los interesados

Implica reconocer personas y organizaciones que serán afectadas por el proyecto y documentar su interés, involucramiento e impacto. Se identifican desde el inicio para desarrollar una estrategia que maximice su influencia positiva y minimice la resistencia.

Registro y clasificación de interesados

- Se evalúan requerimientos, expectativas y fases relevantes.
- Se clasifican como internos o externos, apoyan, son neutrales o se oponen.
- Se registra: nivel deseado de participación, agrupaciones asociadas, impacto potencial y estrategias de atención.

Planear la comunicación

Se determina:

- Quién necesita información
- Qué información necesita
- Cuándo y cómo se entregará
- Quién será responsable de entregarla

Se asignan recursos para asegurar oportunidad, pertinencia y confidencialidad de la información.

Distribuir información

Proceso continuo que hace disponible la información relevante para los interesados, conforme al plan de comunicación.

Técnicas utilizadas:

- Modelos de emisión y recepción
- Selección de medios y estilo
- Reuniones, presentaciones y facilitación

Entregables:

- Reportes, presentaciones, notificaciones, retroalimentación y lecciones aprendidas

---

Administrar las expectativas de los interesados

Proceso que busca influir y atender las necesidades e inquietudes de los interesados mediante:

- Negociación y comunicación continua
- Resolución de problemas y aclaraciones
- Procesamiento de propuestas de cambio

Se apoya en:

- Documentación del proyecto actualizada
- Registro de interesados
- Bitácora de asuntos resueltos

Reportar el desempeño

Consiste en informar sobre el estado del proyecto a través de:

- Reportes de avance
- Mediciones
- Comparaciones entre estimaciones y datos reales

Incluye:

- Análisis del desempeño
- Estado de riesgos y asuntos
- Actividades completadas y planificadas
- Cambios aprobados
- Estimaciones actualizadas de costo y calendario

Introducción al Modelo Relacional

Propuesto por Codd en 1970, el modelo relacional se basa en la teoría de conjuntos. Representa los datos como relaciones (tablas), con el objetivo de mantener independencia lógica y física, simplicidad y flexibilidad en el manejo de datos. El modelo introdujo un enfoque matemático a los DBMS.

Objetivos del Modelo Relacional

- **Independencia física:** los cambios en el almacenamiento físico no afectan el acceso lógico a los datos.
- **Independencia lógica:** los cambios en la estructura lógica no afectan a los usuarios.
- **Flexibilidad:** permite representar los datos según las necesidades del usuario.
- **Uniformidad:** presentación uniforme de los datos.
- **Sencillez:** comprensión y uso sencillo para el usuario final.

Relación

Es el objeto fundamental del modelo. Es un subconjunto del producto cartesiano de varios dominios. Cada relación tiene un grado (número de columnas) y se compone de tuplas (filas). Las relaciones pueden representarse en forma de tabla.

Intensión y extensión de una relación

- **Intensión (esquema):** define los atributos y dominios de la relación.
- **Extensión (instancia):** conjunto de tuplas actuales en la relación.

Ejemplo:

Intensión: AUTOR(NOMBRE:Nombres, NACIONALIDAD:Nacionalidades, INSTITUCION:Instituciones)
Extensión: conjunto de tuplas con valores reales de autores.

Dominio y atributo

- **Dominio:** conjunto de valores atómicos y homogéneos (Ej. edades, colores, DNI).
- **Atributo:** papel que desempeña un dominio dentro de una relación. Pueden ser simples o múltiples.

Llave primaria

Es una columna o conjunto de columnas cuyos valores identifican unívocamente cada tupla. Puede ser:

- **Llave simple:** formada por una sola columna.
- **Llave compuesta:** formada por varias columnas.
- **SuperIlave:** contiene una llave.
- **Llave candidata:** conjunto mínimo de columnas que identifican una tupla.
- **Llave primaria:** una llave candidata elegida para identificar las tuplas.
- **Llaves alternas:** las llaves candidatas no seleccionadas como primarias.

Llave foránea

Es un conjunto de atributos en una relación cuyos valores coinciden con la clave primaria de otra relación. Permite establecer vínculos entre tablas.

Ejemplo:

- Empleados(departamento, puesto) tiene llaves foráneas hacia Departamentos y Puestos.

Puede contener valores nulos si no es parte de una llave primaria en su relación.

Modelo relacional de una base de datos

Es un conjunto de esquemas de relaciones conectadas mediante llaves foráneas, con un propósito común (información de una empresa o proceso).

Restricciones en el modelo relacional

Restricciones inherentes:

- No hay tuplas duplicadas.
- El orden de las tuplas y columnas no importa.
- Cada atributo toma un valor atómico.
- Ningún atributo de la clave primaria puede ser nulo.

Restricciones de usuario:

- **Integridad referencial:** una clave foránea debe coincidir con la clave primaria correspondiente o ser nula.
- Opciones ante operaciones:
  - **Restringida:** impide borrar si hay dependencia.
  - **Cascada:** elimina o modifica registros relacionados.
  - **Poner a nulo:** pone nulo el valor foráneo.
  - **Valor por defecto:** asigna un valor predeterminado.
  - **Procedimiento de usuario:** ejecuta lógica definida por el usuario.

Modelo relacional y arquitectura ANSI

- **Nivel conceptual:** dominios, relaciones, claves y restricciones.
- **Nivel externo:** vistas, definidas sobre tablas base.
- **Nivel interno:** no está especificado en el modelo lógico relacional.

Valores nulos

Representan información desconocida o no aplicable. Son necesarios cuando:

- Se desconoce un valor.
- Se agrega un nuevo atributo a una tabla existente.
- El atributo no aplica a todas las tuplas.

Dinámica del modelo relacional

Los cambios se expresan mediante lenguajes de manipulación relacional:

- **Álgebra relacional:** usa operadores que reciben y generan relaciones.
- **Cálculo relacional:** usa predicados para describir el estado deseado (orientado a tuplas o dominios).

Álgebra relacional

Es una estructura matemática con un conjunto (relaciones) y operadores que generan nuevas relaciones (álgebra cerrada). Los operadores permiten realizar consultas y transformaciones sobre las relaciones.

Operadores del álgebra relacional

- **Unión:** combina tuplas distintas de dos relaciones compatibles.
- **Intersección:** obtiene tuplas comunes a ambas relaciones.
- **Diferencia:** tuplas de una relación que no están en la otra.
- **Proyección:** selecciona columnas específicas de una relación.
- **Selección:** selecciona filas que cumplen una condición.
- **Producto cartesiano:** combina todas las tuplas posibles de dos relaciones.
- **Join natural:** une relaciones por columnas comunes con igual valor.
- **Teta-Join:** une relaciones con una condición explícita, no requiere columnas comunes.
- **División:** obtiene tuplas que cumplen una condición para todos los valores de otra relación.

Observaciones sobre operadores

- Unión e intersección son conmutativas; la diferencia no lo es.
- Los operadores se aplican a relaciones con el mismo esquema de atributos.
- El producto cartesiano puede generar grandes cantidades de datos; se usa como base para otros operadores como el join.
- La división es útil para consultas universales como "elementos que cumplen cierta condición con todos los valores de otro conjunto".

Diagramas de secuencia con UML

Un diagrama de interacción muestra un conjunto de objetos, sus relaciones y los mensajes que intercambian.
Cuando se destaca el orden temporal, se denomina **diagrama de secuencia**.
Cuando se resalta la estructura, se llama **diagrama de colaboración**.

Diagramas de comportamiento - Diagramas de secuencia

- Modelan la interacción entre objetos en un sistema.
- También se conocen como *sequence diagrams*, *event-trace diagrams* o *event scenarios*.
- Tienen dos dimensiones:
  - **Vertical**: representa el tiempo.
  - **Horizontal**: representa las instancias u objetos.

Utilidad de los diagramas de secuencia

- Se usan para describir cómo interactúan objetos en una aplicación a lo largo del tiempo.
- Se modela un diagrama por cada caso de uso.
- Permiten detallar la implementación del escenario, incluyendo:
  - Objetos involucrados
  - Clases relacionadas
  - Mensajes enviados
- A diferencia del diagrama de casos de uso, que modela una vista de negocio, el de secuencia da un **detalle técnico**.

Elementos básicos

Línea de vida

- Línea vertical discontinua que representa la existencia de un objeto en el tiempo.

Foco de control

- Rectángulo delgado sobre la línea de vida.
- Indica el periodo en el que el objeto está realizando una acción.

Creación y destrucción de objetos

- `<<create>>` indica la creación de un objeto.
- `<<destroy>>` indica la destrucción de un objeto.
- Ambos estereotipos son **opcionales** en la notación.

Tipos de mensajes

- **Síncrono**: llamada a un método; el emisor espera la respuesta.
- **Asíncrono**: no bloquea al emisor; puede crear un nuevo hilo o continuar la ejecución.
- **Retorno**: respuesta a un mensaje; puede omitirse si es evidente.

Tipos de diagramas

Diagrama de instancia

- Representa un escenario específico (una ejecución concreta del caso de uso).

Diagrama genérico

- Representa toda la lógica del caso de uso.
- Incluye:
  - Ramificaciones
  - Condiciones
  - Bucles

Fragmentos combinados (operadores)

Se usan para modelar estructuras de control en el diagrama.
Se encierran en un marco con el nombre del operador.

`alt` (alternativa)

- Equivale a una estructura **if…then…else**.

`opt` (opción)

- Representa una condición opcional, como un **if** con una sola rama.

`loop` (bucle)

- Representa iteraciones.
- Incluye una **guarda** que determina cuántas veces se repite.

`sd` (sequence diagram)

- Encierra **todo el diagrama de secuencia**.

`ref` (referencia)

- Hace referencia a otra interacción.
- Se puede usar para:
  - Reutilizar secuencias
  - Encapsular lógica compleja
  - Enviar parámetros y recibir retornos

`par` (paralelo)

- Fragmentos dentro del marco se ejecutan en **paralelo**.

`critical` (región crítica)

- Fragmento que solo puede ser ejecutado por **un proceso a la vez**.

SQL y Álgebra Relacional

¿Qué es SQL?

SQL (Structured Query Language) es el lenguaje que utilizan los sistemas de gestión de bases de datos (DBMS) para definir, consultar y manipular datos.
Su funcionamiento está basado en conceptos del álgebra relacional.

Estructura de una consulta SQL

Una consulta SQL está compuesta por tres cláusulas principales:

- **SELECT**: especifica las columnas que se desean obtener (proyección).
- **FROM**: indica las tablas de donde se obtienen los datos (relaciones).
- **WHERE**: establece condiciones para filtrar las filas (selección).

Relación entre álgebra relacional y SQL

El álgebra relacional es un lenguaje formal que define cómo operar sobre relaciones (tablas).
SQL permite expresar esas operaciones utilizando sentencias declarativas.
Cada operador del álgebra relacional tiene una forma de escribirse en SQL.

Notación del álgebra relacional

- **SL{condición}**: selección
- **PR{columnas}**: proyección
- **JN**: reunión natural (join)
- **JN{condición}**: reunión con condición (theta join)
- **UN**: unión
- **IN**: intersección
- **-**: diferencia
- **X**: producto cartesiano

Esquema de referencia

Los siguientes son ejemplos de tablas usadas para ilustrar equivalencias:

- Materiales (Clave, Descripción, Precio)
- Proveedores (RFC, Razón Social)
- Proyectos (Número, Denominación)
- Entregan (Clave, RFC, Número, Fecha, Cantidad)

Equivalencias entre operaciones

- La consulta de todos los datos de una tabla corresponde a obtener la relación completa.
- La selección permite obtener filas que cumplen cierta condición (por ejemplo, clave igual a un valor).
- La proyección permite seleccionar solo algunas columnas específicas de la tabla.
- La reunión natural une dos relaciones cuando comparten atributos con valores iguales.
- La reunión con condición permite especificar un criterio de emparejamiento diferente al natural.
- La unión obtiene las tuplas de dos relaciones que comparten el mismo esquema.
- La intersección obtiene solo las tuplas que existen en ambas relaciones.
- La diferencia devuelve las tuplas de una relación que no están en otra.
- El producto cartesiano combina todas las tuplas de una relación con todas las de otra.

Funciones agregadas en SQL

Las funciones agregadas permiten resumir datos y obtener resultados globales sobre columnas numéricas:

- **SUM**: suma total de los valores.
- **AVG**: promedio de los valores.
- **MIN**: valor mínimo.
- **MAX**: valor máximo.
- **COUNT**: número total de registros (o de valores no nulos).
- **STD**: desviación estándar.

Agrupaciones

Cuando se desea obtener resultados agregados por grupos (por ejemplo, por producto o por fecha), se usa una agrupación:

- Se define una o varias columnas por las cuales agrupar los datos.
- Se aplican funciones agregadas sobre cada grupo.
- Opcionalmente se puede usar una condición adicional sobre los grupos mediante la cláusula HAVING (por ejemplo, filtrar solo aquellos con ventas mayores a cierto monto).

Ejemplos comunes de agrupación

- Total de ventas por producto.
- Total de ventas por producto y por día.
- Total de ventas por cliente y día, incluyendo suma, promedio, mínimos y máximos, con filtro por importe.
- Totales generales de toda la tabla sin agrupar.

Generación de Casos de Prueba a partir de Casos de Uso

Importancia del testing en el desarrollo de software

- Las pruebas representan entre el 30% y 50% del costo total del desarrollo.
- A pesar de ese gasto, muchos creen que el software no se prueba adecuadamente.
- Dos razones principales:
  - Probar software es difícil.
  - Generalmente se hace sin una metodología clara.
- RUP recomienda empezar las pruebas lo antes posible.
- Un plan claro de pruebas mejora la cobertura, eficiencia y calidad.

Ventajas de usar Casos de Uso para las pruebas

- Permite iniciar las pruebas en fases tempranas.
- Alinea las pruebas con los requisitos funcionales.
- Mejora la cobertura de pruebas.
- Permite una mejor metodología para el diseño de casos de prueba.

Definiciones clave

Caso de Uso

- Define requisitos funcionales desde la perspectiva del usuario.
- Describe una secuencia de acciones con valor observable.
- Sirve como base para desarrollo, pruebas, documentación y validación.

Caso de Prueba

- Define condiciones, entradas y resultados esperados para probar el sistema.
- Verifica la implementación exitosa de los requisitos.

Estructura textual de un caso de uso

- **Nombre**: Identificador del caso.
- **Descripción breve**: Propósito del caso.
- **Flujo de eventos**:
  - Flujo básico: lo que ocurre normalmente.
  - Flujos alternos: excepciones, errores, variaciones.
- **Requisitos especiales**: Requisitos no funcionales.
- **Precondiciones**: Estado del sistema antes de comenzar.
- **Postcondiciones**: Estado del sistema al finalizar.

Escenarios

- Cada escenario es un camino completo a través del caso de uso.
- Pueden combinar el flujo básico con uno o más flujos alternos.

Proceso para generar casos de prueba

Paso 1: Generar escenarios

- Identificar todas las combinaciones posibles de flujo básico y flujos alternos.
- Documentarlos en una matriz de escenarios.

Paso 2: Identificar casos de prueba

- Crear al menos un caso de prueba por cada escenario.

- Verificar que haya pruebas para condiciones normales y excepcionales.
- Incluir condiciones válidas (V), inválidas (I), o no aplicables (N/A).

Paso 3: Asignar valores de datos

- Reemplazar V/I/N/A con valores reales.
- Permite ejecutar las pruebas en la implementación.
- Validar que cubren todas las condiciones relevantes.

Beneficios del enfoque

- Anticipa la detección de errores antes del código.
- Facilita la planificación, seguimiento y cobertura total de pruebas.
- Simplifica el trabajo de desarrollo y asegura que el sistema cumpla requisitos.

Conclusión

El uso de casos de uso para generar casos de prueba mejora la calidad del software, reduce costos al detectar errores antes, y proporciona una metodología clara para validar que el sistema cumple con lo esperado por el usuario y otros sistemas involucrados.

Consultas con Roles y Subconsultas en SQL

Consultas con Roles

Problema

En ocasiones una misma tabla debe ser utilizada varias veces dentro de una consulta SQL, para representar diferentes roles o para comparar subconjuntos de la misma.

Ejemplo: Viajes entre ciudades

Tablas derivadas:

```
viajes(idviaje, idorigen, iddestino, fecha)
ciudades(idciudad, nombreciudad)
```

Consulta deseada: identificador del viaje, nombre de ciudad origen, nombre de ciudad destino y fecha.

Solución con Alias:

```
SELECT idviaje, origen.nombre, destino.nombre, fecha
FROM viajes, ciudades origen, ciudades destino
WHERE viajes.idorigen = origen.idciudad
  AND viajes.iddestino = destino.idciudad;
```

Alternativa con Sinónimos:

```
CREATE SYNONYM origen FOR ciudades;
CREATE SYNONYM destino FOR ciudades;

SELECT idviaje, origen.nombre, destino.nombre, fecha
FROM viajes, origen, destino
WHERE viajes.idorigen = origen.idciudad
  AND viajes.iddestino = destino.idciudad;
```

Ejemplo: Empleados y sus jefes

Tabla:

```
empleados(idempleado, nombre, idjefe)
```

Consulta: nombre del empleado y nombre del jefe.

Solución con Alias:

```
SELECT e.nombre AS empleado, j.nombre AS jefe
FROM empleados e, empleados j
WHERE e.idjefe = j.idempleado;
```

Solución con Sinónimo:

```sql
CREATE SYNONYM jefes FOR empleados;

SELECT empleados.nombre AS empleado, jefes.nombre AS jefe
FROM empleados, jefes
WHERE empleados.idjefe = jefes.idempleado;
```

Ejemplo: Diferencia de ventas entre fechas

Tablas:

```
ventasdiarias(idproducto, fecha, cantidad)
productos(idproducto, descripción, precio)
```

Consulta: Diferencia de ventas entre el 1/SEP/2000 y el 2/SEP/2000 por producto.

Solución con Alias:

```sql
SELECT p.idproducto, p.descripcion,
       vprimero.cantidad - vsegundo.cantidad AS diferencia
FROM productos p,
     ventasdiarias vprimero,
     ventasdiarias vsegundo
WHERE p.idproducto = vprimero.idproducto
  AND p.idproducto = vsegundo.idproducto
  AND vprimero.fecha = '1-SEP-00'
  AND vsegundo.fecha = '2-SEP-00';
```

Subconsultas

Caso 1: Productos no vendidos

Tablas:

```
ventasdiarias(idproducto, fecha, cantidad)
productos(idproducto, descripcion, precio)
```

a) Usando diferencia:

```sql
SELECT idproducto FROM productos
MINUS
SELECT idproducto FROM ventasdiarias;
```

b) Usando NOT IN:

```sql
SELECT idproducto
FROM productos
WHERE idproducto NOT IN (SELECT idproducto FROM ventasdiarias);
```

c) Usando NOT EXISTS:

```sql
SELECT idproducto
FROM productos p
WHERE NOT EXISTS (
  SELECT * FROM ventasdiarias v
  WHERE v.idproducto = p.idproducto
);
```

Caso 2: Productos con ventas mayores a $1,000,000

Consulta con subconsulta anidada:

```sql
SELECT idproducto, descripcion
FROM productos p
WHERE 100000 < (
  SELECT SUM(v.cantidad * p.precio)
  FROM ventasdiarias v
  WHERE v.idproducto = p.idproducto
);
```

Sintaxis General para Subconsultas en SQL

```sql
SELECT columnas
FROM tablas
WHERE condicion_subconsulta;
```

Alternativas para condición:

```
expresión [NOT] IN (SELECT columnas FROM tablas WHERE condiciones)

expresión {> | < | = | != | <> | >= | <=} [{ANY | SOME | ALL}]
(SELECT columnas FROM tablas WHERE condiciones)

[NOT] EXISTS
(SELECT * FROM tablas WHERE condiciones)
```

Conclusión

El uso de alias y sinónimos facilita consultas complejas que requieren que una misma tabla cumpla diferentes roles. Las subconsultas permiten resolver problemas de pertenencia, existencia y cálculos anidados de forma clara y estructurada, mejorando la expresividad del lenguaje SQL.

Normalización de Bases de Datos

Introducción

La normalización busca diseñar estructuras lógicas y estables que eviten anomalías de almacenamiento, redundancias e inconsistencias. Un buen diseño permite:

- Mayor esperanza de vida de la base de datos.
- Flexibilidad ante nuevos requerimientos.
- Buen desempeño a medida que aumenta el volumen de datos.

Razones para normalizar

- Representar relaciones relevantes entre datos.
- Facilitar la consulta y generación de reportes.
- Simplificar el mantenimiento (actualizaciones, inserciones, eliminaciones).
- Minimizar la necesidad de reestructurar la base ante nuevas aplicaciones.

Proceso de Normalización

1. **Descomponer registros complejos en tablas bidimensionales.**
2. **Eliminar dependencias no completas respecto a la llave primaria.**
3. **Eliminar dependencias transitivas.**

Las "formas normales" son los niveles que indican qué restricciones cumple una tabla.

Primera Forma Normal (1FN)

Una tabla está en 1FN si:

- Todos los valores son **atómicos** (un solo valor por celda).
- Cada columna contiene valores del **mismo tipo**.
- No existen grupos repetidos ni arreglos.
- Cada columna tiene nombre único.
- No hay filas duplicadas.

Ejemplo:

Alumno (Control, Nombre, Esp)

Materia (Clave, NomM, Creditos)

Ambas cumplen con 1FN al tener valores simples y del mismo tipo.

Segunda Forma Normal (2FN)

Una tabla está en 2FN si:

- Está en 1FN.
- Todos los atributos no clave **dependen completamente** de la llave primaria (no parcial).

> Si una tabla tiene llave simple, está automáticamente en 2FN.

Se eliminan **dependencias parciales**.

Tercera Forma Normal (3FN)

Una tabla está en 3FN si:

- Está en 2FN.
- Todos los atributos no clave **no dependen transitivamente** de la llave primaria.

> Una dependencia transitiva ocurre si A → B y B → C, pero A no → C directamente.

Se eliminan **atributos que dependen de otros atributos no clave**.

Forma Normal de Boyce-Codd (BCNF)

Una tabla está en BCNF si:

- Cada **determinante** es una **llave candidata**.

> Las llaves candidatas son conjuntos mínimos de atributos que identifican únicamente a cada fila.

BCNF es una versión más estricta que la 3FN.

Cuarta Forma Normal (4FN)

Una tabla está en 4FN si:

- Está en BCNF.
- No contiene **dependencias de valores múltiples**, salvo que:
  - Sean triviales.
  - La parte izquierda de la dependencia sea una superllave.

> Una dependencia de valores múltiples ocurre cuando un atributo A determina varios valores de B y C, pero B y C son **independientes entre sí**.

Ejemplo:

Estudiante (Clave, Especialidad, Curso)

Separado en:

- **Especialidades (Clave, Especialidad)**
- **Cursos (Clave, Curso)**

Quinta Forma Normal (5FN)

Una tabla está en 5FN si:

- Está en 4FN.
- Todas las **dependencias de producto** cumplen:
  - Ser triviales.
  - Cada subconjunto es una superllave.

> 5FN trata casos donde una tabla puede descomponerse en subtablas, pero no puede reconstruirse sin perder información.
